Statistical data analysis

Learning from Sequential Genomic Data

A biologically-directed final assignment

Fall 2021/2022

Introduction

You are provided with three sets of DNA sequences. The first set represents introns. The other two sets represent random DNA sequences that only look similar, they have the same beginning and the end as the introns have. The goal is to learn from the sequential data, in particular, to understand relationships between the DNA structure and its classification.

1 Background

Hereditary information encoding the development and functioning of an organism is stored in a macromolecule called *deoxyribonucleic acid* (DNA). The information is stored as a sequence of nucleotides also called *bases*, namely adenine (A), cytosine (C), guanine (G) and thymine (T). The information carried by DNA is held in the sequence of distinguishable regions of DNA called *genes*. *Introns* and *exons* are continuous nucleotide sequences within a gene. An intron (an acronym for intragenic region) is any nucleotide sequence within a gene that is removed by RNA splicing during maturation of the final RNA product. In other words, introns are non-coding regions of an RNA transcript, or the DNA encoding it, that are eliminated by splicing before translation. Sequences that are joined together in the final mature RNA after RNA splicing are exons. They code for amino acids and proteins. An *intergenic region* is a stretch of DNA sequence located between genes. Intergenic regions are noncoding and may have regulatory function.

In the human genome, introns are much longer than exons. They can make up as much as 90% of a gene and can be over 10,000 nucleotides long. Introns are prevalent in genes; over 90% of human genes contain introns with an average of nine introns per gene. An intron is a stretch of DNA that begins and ends with a specific series of nucleotides. These sequences act as the boundary between introns and exons and are known as *splice sites*. The recognition of the boundary between coding and non-coding DNA is crucial for the creation of functioning genes.

In this task you will deal with genomes of simple eukaryotes. Their introns are much shorter than human ones mentioned above, their average length is around 60 nucleotides. In our intron file they always begin with 5‘ GT (the donor splice site) and end in AG 3‘ (the acceptor splice site). The referential negative sequences are random parts of the same genomes having the following characteristics: they start and end with the same splice sites as introns do and have approximately the same length, they are taken from an arbitrary part of genome (intergenic, exons, overlaps between introns and exons, etc.). This means

that the information about the length, the beginning and the end cannot directly be used in intron recognition. However, there are other conserved sequences found in introns. The main goal of this task is to understand the intron structure which should help to recognize introns in DNA sequences.

2 Data

You are given three FASTA files. The file *posIntron.fasta* contains 181,968 intron sequences taken from genomes of several different species. In further text we will denote these sequences as introns. The file *negIntron.fasta* contains the same number of random sequences taken from the same pool of genomes (as already mentioned above, they are not completely random, remember the length, the GT beginning and the AG end). In further text, these sequences will be referred to as random. The file *negIntronWithSSites.fasta* contains the same number of negative sequences again. The sequences have the same characteristics as in *negIntron.fasta* (they do not stand for introns, but have similar lengths and start and stop with the same dinucleotides as introns do). Unlike the previous file, their splice site neighborhood looks similar to neighborhoods of true intron splice sites (assessed by an unknown but reliable computational model). The sequences from the third file will be named as splicesites.

The FASTA structure is simple. It is a text file, every sequence is represented by two lines. The description line begins with > and gives a name for the sequence. It may also contain additional information about it. In our files it simply numbers the sequence that follows. Following the header line, the actual sequence is provided. The nucleic acid codes supported in FASTA are: A, C, G, T, but also the degenerate codes such as N (any nucleic acid symbol), R (A or G) and several others could in general be used. The degenerate symbols represent for a position on a DNA sequence that can have multiple possible alternatives. However, the degenerate symbols do not appear in our files.

```
>seq_1
GTGCGAATGGAGTAGAGCGCGTCCTTGCTGCAAGCTCACTGACTCGTCAATGTTCAG
>seq_2
GTGAGTGCATGGGATCTGAAGTCCTGTCGGCTCGCTCAATCTCGGTCGGCCAAAG
```

Fig. 1 A FASTA file, a few header lines representing two introns. The donor sites (GT) in blue, the acceptor sites (AG) in red.

3 Tasks

The assignment can be decomposed into the following well-defined tasks:

1. **Preprocessing**. Load the FASTA files and turn the sequences into numeric vectors. The simplest way to do it is to use k-mer (relative) frequency. In bioinformatics, k-mers are subsequences of length k contained within a biological sequence. They are primarily used within the context of computational genomics and sequence analysis. In R, you can use *ape* and *kmer* libraries, their application is shown in *preprocess.R*. To illustrate the decomposition of a sequence into a k-mer frequency vector, see the following example:

```
GTGGGAATGG, k=3 ->  GTG 1, TGG 2, GGG 1, GGA 1, GAA 1, AAT 1, ATG 1.
```

Fig. 2 The decomposition of a sequence into its k-mer frequency vector.

The main parameter to optimize in this subtask is the value of k. Short k-mers tend to lose the structural information contained in the sequence while long k-mers could lead to sparse and long frequency vectors that are difficult to learn from. Of course, you can also propose and implement preprocessing that goes beyond the k-mers, for example a search for structural patterns that frequently appear in introns and rarely appear in random sequences, but this is not expected in any case and would be a time-consuming bonus task.

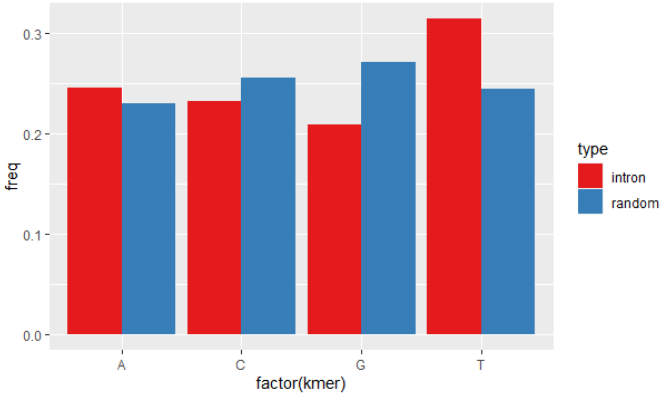
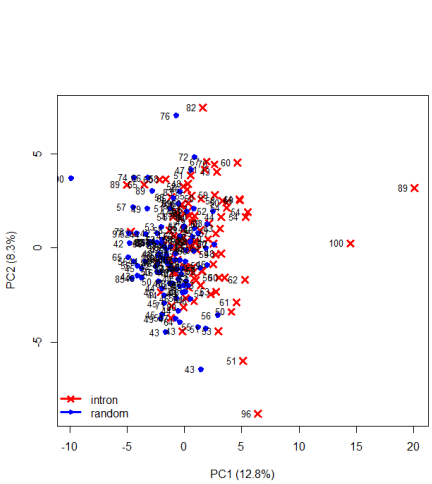


Fig. 3 The difference between introns and random sequences in terms of the relative frequency of bases. The graph suggests that even the most simple 1-mers may help to distinguish between the sequence types. The relative frequency of bases is not equal in introns and random sequences.

2. **Exploratory analysis of the frequency vectors**. Take the frequency file developed in the previous step, carry out dimensionality reduction and clustering. The main goal is to see whether the class (intron/splicesite/random) manifests in clusters. Show the explanatory plots and explain the relationship between the observed sample distribution and the expected distribution. Let us assume that sequences from the same group should have similar frequency profiles, different species may also form different clusters (however, the name of species is not available in the sequence annotation). An expected result could look similar to following (run on downsampled data, splicesites completely omitted, sequence lengths shown in the PCA plot):



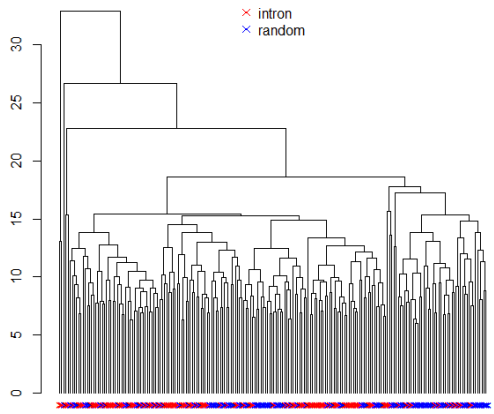


Fig. 4 Principal component analysis Hierarchical clustering, dendrogram

3. **Differential k-mers.** Find the k-mers whose relative frequency statistically differs among the sequence types. Select a proper statistical method (ideas: t-test, ANOVA and their non-parametric counterparts), test their assumptions, do not forget to do multiple testing correction. An example of potentially differential k-mer could be the 1-mer T (see Fig.3). Do not forget to generalize for longer k-mers. You do not have to find all the possible differential k-mers, however, provide a systematic identification method.

4. **Predictive model.** Create a model that predicts the sequence class (intron/splicesite/random) based on the information contained in sequence. Definitely consider linear/quadratic discriminant analysis and logistic regression, but you can try any other learning algorithm too. Evaluate and compare the performance of the models. Utilize feature selection to simplify the models and improve their performance. The numbers of sequences are large, cross-validation is not necessary here, you can employ the hold-out method instead (a simple train/test/validate split), but use the sets properly. You do not have to use all the available samples if necessary.

5. **Skewed classes.** The number of introns in real genomes is much smaller than the number of random and splicesite sequences. In other words, our files downsample the genomes with different sampling rates. Consider that the true number of introns is 50 times smaller than the number of random sequences and 10 times smaller than the number of splicesite sequences. Explain how this information changes your solution in the previous steps.

4 Submission and evaluation

Submit your solution to the upload system. Submit only the rmarkdown file named intronSequences$YOURFELUSERNAME.Rmd. This file should be considered as a report containing a definition of the task, description of your implementation details, graphical outcomes and your **detailed answers** to the required tasks. Remark: a Python notebook can be submitted instead.

You can obtain up to 15 points for this assignment. The subtasks will be scored as follows: 1 point for preprocessing, 3 points for exploratory analysis, 3 points for differential k-mers, 5 points for prediction and 2 points for class skewness. Approximately 60% will be given for the concept of the solution (selection of statistical methods and their correct application, depth of the solution), 30% for the answers that summarize your solution in the individual subtasks (interpretation of your results and explanation of their practical impact in natural language), 10% for formal issues (clarity of the code, comments, readability of the rmarkdown as a whole).